MEMORY-EFFICIENT SYSTEM FOR DECISION TREE MACHINE LEARNING

FIELD OF THE SPECIFICATION

[0001] This disclosure relates in general to the field of artificial intelligence and machine learning, and more particularly, though not exclusively, to a memory-efficient system for decision tree machine learning.

BACKGROUND

[0002] Tree data structures, such as binary trees and decision trees, have a wide variety of applications in computer science, from data storage and searching to artificial intelligence and machine learning. These data structures can be quite memory inefficient, however, particularly when stored on fixed-memory architectures that require memory to be statically allocated before runtime. For example, since the actual size of each tree node is not known until runtime, the amount of pre-allocated memory must be large enough to accommodate the maximum possible size for each tree node, even though most or all nodes will not use that much memory.

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] The present disclosure is best understood from the following detailed description when read with the accompanying figures. It is emphasized that, in accordance with the standard practice in the industry, various features are not necessarily drawn to scale, and are used for illustration purposes only. Where a scale is shown, explicitly or implicitly, it provides only one illustrative example. In other embodiments, the dimensions of the various features may be arbitrarily increased or reduced for clarity of discussion.

[0004] FIG. **1** illustrates a schematic diagram of an example computing system in accordance with certain embodiments.

[0005] FIG. **2** illustrates an example of data discretization.

[0006] FIG. **3** illustrates a block diagram for an example embodiment of optimized data discretization.

[0007] FIG. **4** illustrates a flowchart for an example embodiment of optimized data discretization.

[0008] FIGS. **5**A-E provide a comparison of various data discretization approaches in a variety of use cases.

[0009] FIG. **6** illustrates an example embodiment of an electronic device with data discretization functionality.

[0010] FIG. **7** illustrates an example embodiment of an edge device with an optimized decision tree machine learning (ML) engine.

[0011] FIGS. **8**A-B illustrate an overview of a random forest machine learning (ML) algorithm.

[0012] FIGS. **9**A-C illustrate an example of using automated data binning to compute feature value checkpoints for training a decision tree model.

[0013] FIG. **10** illustrates a process flow for efficiently training a random forest machine learning (ML) model in accordance with certain embodiments.

[0014] FIG. **11** illustrates an example embodiment of an artificial intelligence (AI) accelerator implemented with an optimized decision tree machine learning (ML) engine.

[0015] FIGS. **12**A-G illustrate a performance comparison of an optimized random forest versus a traditional random forest.

[0016] FIG. **13** illustrates a flowchart for performing decision tree training and inference in accordance with certain embodiments.

[0017] FIG. **14** illustrates the structure of a decision tree in a typical random forest machine learning classifier.

[0018] FIG. **15** illustrates an example implementation of a tree data structure on a fixed-memory hardware architecture.

[0019] FIG. **16** illustrates a memory-efficient implementation of a tree data structure.

[0020] FIGS. **17**A-B illustrate a memory usage comparison for various implementations of tree data structures.

[0021] FIG. **18** illustrates a flowchart for a memory-efficient implementation of decision tree machine learning in accordance with certain embodiments.

[0022] FIG. **19** illustrates an overview of an edge cloud configuration for edge computing.

[0023] FIG. **20** illustrates operational layers among endpoints, an edge cloud, and cloud computing environments.

[0024] FIG. **21** illustrates an example approach for networking and services in an edge computing system.

[0025] FIG. **22** illustrates a compute and communication use case involving mobile access to applications in an edge computing system.

[0026] FIG. **23**A provides an overview of example components for compute deployed at a compute node in an edge computing system.

[0027] FIG. **23**B provides a further overview of example components within a computing device in an edge computing system.

[0028] FIG. **24** illustrates an example software distribution platform to distribute software to one or more devices in accordance with certain embodiments.

EMBODIMENTS OF THE DISCLOSURE

[0029] The following disclosure provides many different embodiments, or examples, for implementing different features of the present disclosure. Specific examples of components and arrangements are described below to simplify the present disclosure. These are, of course, merely examples and are not intended to be limiting. Further, the present disclosure may repeat reference numerals and/or letters in the various examples. This repetition is for the purpose of simplicity and clarity and does not in itself dictate a relationship between the various embodiments and/or configurations discussed. Different embodiments may have different advantages, and no particular advantage is necessarily required of any embodiment.

Optimized Data Discretization and Binning

[0030] Data analytics has a wide range of applications in computing systems, from data mining to machine learning and artificial intelligence, and has become an increasingly important aspect of large-scale computing applications. Data preprocessing, an important initial step in data analytics, involves transforming raw data into a suitable format for further processing and analysis. For example, real-world or raw data is often incomplete, inconsistent, and/or error prone. Accordingly, raw data may go through a series of preprocessing steps, such as data cleaning, integration, transformation, reduction, and/or discretization or quantization. Data discretization, for example, may involve converting or partitioning a range of continuous raw data into a smaller number of intervals or values. For example, data